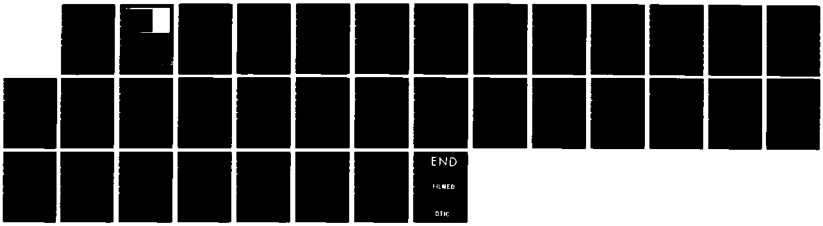
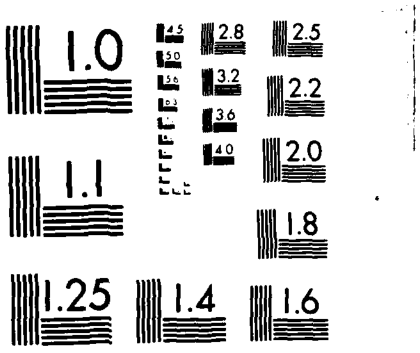
AD-A153 476   A QUADRATIC PROGRAMMING ALGORITHM(U) WISCONSIN UNIV-MADISON MATHEMATICS RESEARCH CENTER M J BEST ET AL. DEC 84 MRC-TSR-2776 DAAG29-80-C-0041   1/1

UNCLASSIFIED   F/G 12/1   NL




1.0
4.5
5.0
5.6
6.3
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

(2)

AD-A153 476

MRC Technical Summary Report #2776

A QUADRATIC PROGRAMMING ALGORITHM

M. J. Best and K. Ritter

Mathematics Research Center
University of Wisconsin—Madison
610 Walnut Street
Madison, Wisconsin 53705

December 1984

(Received August 20, 1984)

Sponsored by

U. S. Army Research Office
P. O. Box 12211
Research Triangle Park
North Carolina 27709

UNIVERSITY OF WISCONSIN - MADISON
MATHEMATICS RESEARCH CENTER

A QUADRATIC PROGRAMMING ALGORITHM

M. J. Best* and K. Ritter**

Technical Summary Report #2776
December 1984

ABSTRACT

By using conjugate directions a method for solving convex quadratic programming problems is developed. The algorithm generates a sequence of feasible solutions and terminates after a finite number of iterations. Extensions of the algorithm for nonconvex and large structured quadratic programming problems are discussed.

AMS(MOS) Subject Classification: 90C20, 90C25.

Key Words: Quadratic programming, optimization, conjugate directions, *decompostion*.

Work Unit Number 5 - Optimization and Large Scale Systems

* Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 361.
** Institut für Angewandte Mathematik und Statistik, Technische Universität München, Arcisstrasse 21, 8000 München 2, West Germany.

Sponsored by the United States Army under Contract No. DAAG29-80-C-0041 and in part by the Natural Sciences and Engineering Research Council of Canada under Grant Nos. A 81 89 and E 55 82.

SIGNIFICANCE AND EXPLANATION

The quadratic programming problem is the following: Given $n \times 1$ vectors $c, a_1, \ldots, a_m$, numbers $b_1, \ldots, b_m$ and an $n \times n$ matrix C, find an $n \times 1$ vector x which minimizes the quadratic function

$$c'x + \frac{1}{2} x'Cx ,$$

subject to the inequality constraints

$$a_i'x \leq b_i , \quad i = 1, \ldots, m.$$

If C is the $n \times n$ zero matrix, then the quadratic programming problem reduces to the linear programming problem.

In recent years quadratic programming has become an important tool in optimization. It has wide applications in areas such as statistics, structural engineering, economics and portfolio analysis.

The contribution of this work is an algorithm which solves the quadratic programming problem in a finite number of steps. Furthermore, an extension of the algorithm is given which can be used to solve large structured quadratic programming problems.

The responsibility for the wording and views expressed in this descriptive summary lies with MRC, and not with the authors of this report.

A QUADRATIC PROGRAMMING ALGORITHM

M. J. Best* and K. Ritter**

1. Introduction

We consider the quadratic programming problem

$$\min\{c'x + \frac{1}{2}x'Cx \mid a_i'x \le b_i,\ i = 1,\dots,m\}\ , \qquad (1)$$

where c, x, $a_1,\dots,a_m$ are n - vectors, C is an (n,n) symmetric matrix and $b_1,\dots,b_m$ are scalars. Prime is used to denote transposition. Let $F(x) = c'x + \frac{1}{2}x'Cx$ denote the objective function for (1) and

$$R = \{x \mid a_i'x \le b_i,\ i = 1,\dots,m\}$$

denote the feasible region. x* satisfies the Karush - Kuhn - Tucker conditions for (1) if there are numbers $u_1,\dots,u_m$ satisfying

$$x^* \in R\ ,$$
$$-c - Cx^* = u_1a_1 + \dots + u_ma_m\ , \quad u \ge 0\ ,$$
$$u_i(a_i'x^* - b_i) = 0, \qquad i = 1,\dots,m\ .$$

If C is positive semi - definite, these conditions are both necessary and sufficient for x* to be a global minimizer [1]. Let

$$I(x^*) = \{i \mid a_i'x^* = b_i,\ 1 \le i \le m\}\ .$$

If C is indefinite and in addition to the Karush - Kuhn - Tucker conditions x* also satisfies

$$u_i > 0\ , \quad \text{all } i \in I(x^*)\ ,$$

* Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1.
** Institut für Angewandte Mathematik und Statistik, Technische Universität München, Arcisstrasse 21, 8000 München 2, West Germany.

Sponsored by the United States Army under Contract No. DAAG29-80-C-0041 and in part by the Natural Sciences and Engineering Research Council of Canada under Grant Nos. A 81 89 and E 55 82.

and

$$s'Cs > 0 \text{ for all } s \neq 0, \text{ with } a_i's = 0, \text{ all } i \in I(x^*)$$

then x^* is a strong local minimizer for (1).

We first consider the case when C is positive semi - definite and present an algorithm for the solution of (1) which, in a finite number of steps, determines either an optimal solution or that the problem is unbounded from below. The algorithm is based on a new updating procedure for conjugate directions when the set of active constraints is changed. A general description of the method is given in Section 2 and a detailed formulation is given in Section 3.

In Section 4, we discuss the case of C being indefinite. Provided suitable initial data is used, the algorithm of Section 3 will determine a local minimizer of (1) in a finite number of steps. A procedure is then given which will ensure that the initial data requirement will be met.

In Section 5, we consider the structured quadratic programming problem: Minimize

$$F_1(x_1) + F_2(x_2) + \dots + F_p(x_p) + F_0(y) \tag{2a}$$

subject to

$$a_\nu' x_i + b_\nu' y \leq \beta_\nu ; \quad \nu = m_{i-1} + 1, \dots, m_i \tag{2b}$$

for $i = 1, \dots, p$ and

$$b_\nu' y \leq \beta_\nu ; \quad \nu = m_p + 1, \dots, m . \tag{2c}$$

Since (2) is a special case of (1), it can be solved using the algorithms presented in Sections 3 and 4. However, the number of variables may be large and the computational expense high. Therefore, it is appropriate to develop an algorithm which takes advantage of the structure of (2). This is done in Section 5.

2. General Description of the Algorithm

$\hat{x}$ is a quasi-stationary point for (1) if $\hat{x} \in R$ and $\hat{x}$ is an optimal solution for

$$\min\{F(x) \mid a_i'x = b_i, \text{ all } i \in I(\hat{x})\} .$$

To ensure finite termination, the algorithm determines a sequence of quasi-stationary points having decreasing objective function values.

Let x_j denote the iterate at iteration j. Suppose x_j is not a quasi-stationary point. Assume for simplicity that the first q constraints are active at x_j. We begin by looking for a quasi-stationary point at which only the first q constraints are active. If such a point exists, it is an optimal solution for

$$\min\{c'x + \tfrac{1}{2}x'Cx \mid a_i'x = b_i, \ i = 1,\ldots,q\} . \tag{3}$$

We write this point in the form $x_j - s_j$, where s_j is to be thought of as a search direction. s_j may be determined as follows. Let

$$D_j' = [a_1,\ldots,a_q, \ Cc_{q+1},\ldots,Cc_n] , \tag{4}$$

and assume that

$$D_j^{-1} = [c_1,\ldots,c_q, \ c_{q+1},\ldots,c_n] . \tag{5}$$

D_j^{-1} has columns $c_1,\ldots,c_n$. By definition of the inverse matrix, $c_{q+1},\ldots,c_n$ are a set of normalized conjugate directions which are orthogonal to the gradients of the active constraints (normalized in the sense that $c_i'Cc_i = 1$, $i = q+1,\ldots,n$). Let $g_j = c + Cx_j$ denote the gradient of F at x_j. Define

$$s_j = \sum_{i=q+1}^{n} (g_j'c_i)c_i \ , \qquad (6)$$

and

$$x_{j+1} = x_j - s_j \ .$$

Since $g_{j+1} = g_j - Cs_j$,

$$g_{j+1}'c_k = g_j'c_k - \sum_{i=q+1}^{n} (g_j'c_i)c_i'Cc_k$$

$$= g_j'c_k - g_j'c_k = 0 \ , \quad k = q+1,\ldots,n \ . \qquad (7)$$

Since the columns of D_j' are linearly independent, there are scalars $\lambda_1,\ldots,\lambda_n$ with

$$g_{j+1} = \lambda_1 a_1 + \ldots + \lambda_q a_q + \lambda_{q+1}Cc_{q+1} + \ldots + \lambda_n Cc_n \ .$$

For k = 1,...,n, take the inner product of both sides of the above with c_k. By definition of the inverse matrix,

$$g_{j+1}'c_k = \lambda_k \ .$$

With (7), this implies

$$g_{j+1} = (g_{j+1}'c_1)a_1 + \ldots + (g_{j+1}'c_q)a_q \ .$$

Provided x_{j+1} satisfies the remaining inequality constraints for (1), then it is also a quasi-stationary point for (1). Furthermore, the multipliers for the active constraints are given by

$$u_i = -g_{j+1}'c_i \ , \quad i = 1,\ldots,q \ . \qquad (8)$$

Because $x_j - s_j$ is optimal for (3), s_j is called the Newton direction.

Next suppose that x_j is a quasi-stationary point. Let D_j and D_j^{-1} be as in (4) and (5). Then $g_j'c_i = 0$, $i = q+1,\ldots,n$. Let u_i, $i = 1,\ldots,q$ be defined by (8). If $u_i \geq 0$ for $i = 1,\ldots,q$ then x_j satisfies the Karush-Kuhn-Tucker conditions for (1) and is thus an optimal solution. Otherwise, suppose $u_q < 0$. We proceed by deleting constraint q from the active set. Suppose first that $c_q'Cc_q = 0$. Then we set $s_j = c_q$ and observe that

$$F(x_j - \sigma s_j) = F(x_j) - \sigma g_j's_j \ .$$

Since $u_q = -g_j's_j < 0$, $F(x_j - \sigma s_j)$ is a strictly decreasing linear function of σ. Either $x_j - \sigma s_j$ is feasible for (1) for all $\sigma \geq 0$ in which case (1) is unbounded from below, or for sufficiently large σ, some previously inactive constraint becomes active. Next suppose that $c_q'Cc_q > 0$. Then by definition of the inverse matrix, $(c_q'Cc_q)^{-1}c_q$ together with $c_{q+1},\ldots,c_n$ form a set of normalized conjugate directions which are orthogonal to $a_1,\ldots,a_{q-1}$. From (6), the Newton direction is

$$s_j = (g_j'c_q)(c_q'Cc_q)^{-1}c_q \ .$$

In fact, it is more convenient to use $s_j = c_q$, which is parallel to the Newton direction and account for the scalar $(g_j'c_q)(c_q'Cc_q)^{-1}$ in the stepsize calculation. Thus when either $c_q'Cc_q = 0$, or, $c_q'Cc_q > 0$, it is appropriate to set $s_j = c_q$.

Assuming $x_j - s_j$ is feasible for (1), we wish to modify D_j so that D_{j+1} and D_{j+1}^{-1} are related as in (4) and (5) but without constraint q. An appropriate way to do this is to replace column q of D_j' with

$\hat{d} \equiv (c_q'Cc_q)^{-1/2}Cc_q$. Let the new matrix be denoted by D_{j+1}' and let

$$D_{j+1}^{-1} = [\hat{c}_1,\ldots,\hat{c}_n] .$$

The Sherman - Morrison formula [4] asserts

$$\hat{c}_i = c_i - \frac{\hat{d}'c_i}{\hat{d}'c_q} c_q , \quad \text{for } i = 1,\ldots,n,\ i \neq q .$$

Because $c_{q+1},\ldots,c_n$ are conjugate directions,

$$\hat{c}_i = c_i - \frac{c_q'Cc_i}{c_q'Cc_q} c_q = c_i , \quad \text{for } i = q+1,\ldots,n ;$$

i.e., the normalized conjugate direction columns remain unchanged by the updating. Furthermore, the Sherman - Morrison formula again asserts that

$$\hat{c}_q = (c_q'Cc_q)^{-1/2}c_q .$$

Define

$$\tilde{\sigma} = \begin{cases} \dfrac{g_j's_j}{s_j'Cs_j} , & \text{if } s_j'Cs_j > 0 , \\ +\infty , & \text{if } s_j'Cs_j = 0 . \end{cases}$$

Then $\tilde{\sigma}_j$ is called the optimal stepsize and $F(x_j - \sigma s_j)$ is a strictly decreasing function of σ for $0 \leq \sigma \leq \tilde{\sigma}_j$. So far we have made the assumption that $x_j - \tilde{\sigma}_j s_j \in R$ provided $s_j'Cs_j > 0$. We now suppose that this is no longer the case. Let $\hat{\sigma}_j$ denote the largest value of σ for which $x_j - \sigma s_j \in R$. It is usual to call $\hat{\sigma}_j$ the maximum feasible stepsize. An explicit formula for $\hat{\sigma}_j$ is readily derived:

$$\hat{\sigma}_j = \min\left\{\frac{a_i'x_j - b_i}{a_i's_j} \;\middle|\; \text{all } i = 1,\dots,m \text{ with } a_i's_j < 0\right\} .$$

Since $F(x_j - \sigma s_j)$ is a strictly decreasing function of σ for $0 \le \sigma \le \tilde{\sigma}_j$, it is appropriate to set $x_{j+1} = x_j - \sigma_j s_j$ with $\sigma_j = \min\{\tilde{\sigma}_j, \hat{\sigma}_j\}$.

We continue by assuming that $\sigma_j = \hat{\sigma}_j < \tilde{\sigma}_j$. Let ℓ be such that

$$\hat{\sigma}_j = \frac{a_\ell'x_j - b_\ell}{a_\ell's_j} .$$

Then constraint ℓ, which was inactive at x_j, becomes active at x_{j+1}. An obvious way to proceed is to obtain D_{j+1}' from D_j' by replacing the q-th column with a_ℓ. If a_ℓ is orthogonal to the last $(n-q)$ columns of D_j^{-1}, then it follows from the Sherman-Morrison formula that these conjugate direction columns will be unchanged by the update. However, *there is no reason* to expect orthogonality and the updating would then destroy the conjugate directions.

In Lemma 1, we introduce an updating procedure which circumvents this difficulty. We motivate it as follows. Continuing the above discussion, suppose

$$\begin{aligned} a_\ell'c_{q+1} &\neq 0 , \\ a_\ell'c_i &= 0 , \quad i = q+2,\dots,n ; \end{aligned} \tag{9}$$

i.e., a_ℓ is in fact orthogonal to all but one of the conjugate directions. Suppose D_{j+1}' is obtained from D_j' by replacing column q+1 with a_ℓ. With

$$D_{j+1}' = [a_1,\dots,a_q,a_\ell,Cc_{q+2},\dots,Cc_n] ,$$

the Sherman - Morrison formula asserts that

$$D_{j+1}^{-1} = [\hat{c}_1,\ldots,\hat{c}_q,\hat{c}_{q+1},c_{q+2},\ldots,c_n] ,$$

the point being that the last $n - q - 1$ columns of D_{j+1}^{-1} form a set of normalized conjugate directions which are orthogonal to the gradients of the active constraints as well as a_q. Although constraint q has become inactive, its gradient is still the q-th column of D_{j+1}'. The situation is identical to that when x_j is a quasi - stationary point and constraint q is to be dropped. As previously discussed, we continue by setting $s_{j+1} = \hat{c}_{q+1}$.

Of course, there is no reason to expect $c_{q+1},\ldots,c_n$ to satisfy (9). The critical idea of this section is to replace $c_{q+1},\ldots,c_n$ with a new set of conjugate directions $\hat{c}_{q+1},\ldots,\hat{c}_n$ which do satisfy (9). The construction procedure is based on the following lemma. Note that although we assume in this section that C is both symmetric and positive semi - definite, the lemma requires only the symmetry assumption.

Let e_ν denote the ν-th unit vector.

<u>Lemma 1</u>

Let $P = [p_1,\ldots,p_k]$ be an (n,k) - matrix satisfying $P'CP = I$ and let d be an n - vector satisfying $P'd \neq 0$. Let ν be any integer with $1 \leq \nu \leq k$. Define

$$u = P'd ,$$

$$\theta_2 = \begin{cases} \sqrt{u^Tu} , & \text{if } p_\nu'd \leq 0 , \\ -\sqrt{u^Tu} , & \text{otherwise} , \end{cases}$$

$$\theta_1 = [2(u'u - \theta_2 p_\nu' d)]^{-1/2} ,$$

$$w = \theta_1(\theta_2 e_\nu - u) ,$$

$$\hat{P} = P(I - 2ww') \equiv [\hat{p}_1,\dots,\hat{p}_k] .$$

Then

a) $\hat{p}_i = p_i - 2w_i p, \quad i = 1,\dots,k$, where $p = w_1 p_1 + \dots + w_k p_k$

b) $\hat{P}'C\hat{P} = I$,

c) $d'\hat{p}_i = 0, \quad i = 1,\dots,k$ and $i \neq \nu$,

d) $\text{span}\{\hat{p}_1,\dots,\hat{p}_k\} = \text{span}\{p_1,\dots,p_k\}$.

Proof:

First note that $\theta_2 p_\nu' d \leq 0$ so that θ_1 is well-defined. Let $Q = I - 2ww'$. Then Q is a Householder-matrix with (see e.g. [2])

$$Q'Q = QQ = I \tag{10}$$

and

$$Qu = \|u\| e_\nu . \tag{11}$$

a) By definition of $\hat{P}$,

$$\hat{P} = P - 2(Pw)w' = P - 2pw' ,$$

i.e.

$$\hat{p}_i = p_i - 2w_i p, \quad i = 1,\dots,k.$$

b) By (10),

$$\hat{P}'C\hat{P} = Q'(P'CP)Q = Q'Q = I.$$

c) Using (11) we have

$$d'\hat{P} = d'PQ = u'Q = \|u\| e_\nu ,$$

from which the assertion follows.

$$\nabla F_i(x_i^j) \in \operatorname{span}\{a_\nu | \nu \in J_{ij}\} \tag{17}$$

$$C_i c_{\nu i} = 0 \text{ for every } \nu \text{ with } \alpha_{\nu i} = 0 \tag{18}$$

The first condition is satisfied if x_i^j is a quasi - stationary point for (16). The second condition can be imposed without loss of generality. Indeed, since C_i is a positive semi - definite matrix we have $C_i c_{\nu i} \neq 0$ if and only if $c'_{\nu i} C_i c_{\nu i} > 0$. In this case D_{ij}^{-1} and J_{ij} can be updated as in Step 3.1 of the algorithm resulting in an $\alpha_{i\nu} = -1$.

For i = 1,...,p define

$$M_{ij} = \sum c_{\nu i} b'_{\alpha_{\nu i}} , \tag{19}$$

where the summation is over all ν such that $\alpha_{\nu i} \geq 1$. Then

$$M'_{ij} a_\nu = b_\nu \quad \text{for all } \nu \in J_{ij} \tag{20}$$

and, for every y,

$$a'_\nu(x_i^j + M_{ij}(y^j - y)) + b'_\nu y = \beta_\nu, \qquad \nu \in J_{ij} \tag{21}$$

$$\nabla F_i(x_i^j + M_{ij}(y^j - y)) \in \operatorname{span}\{a_\nu | \nu \in J_{ij}\} . \tag{22}$$

In order to verify (22) observe that

$$\nabla F_i(x_i^j + M_{ij}(y^j - y) = \nabla F_i(x_i^j) + C_i M_{ij}(y^j - y)$$

and by (17), (18), and the properties of D_{ij}^{-1}

$$c'_{\nu i} \nabla F_i(x_i^j) = 0 \quad \text{and} \quad c'_{\nu i} C_i M_{ij} = 0$$

for all ν such that $\alpha_{\nu i} \leq 0$.

5. Decomposition

In this section we develop a decomposition method for the problem (2), which is a generalization of the linear problem studied by Rosen in [3].

For $i = 1,\dots,p$ we assume that x_i is an n_i - vector, y is an n - vector and

$$F_i(x_i) = c_i'x_i + \frac{1}{2}x_i'Cx_i, \quad F_0(y) = c'y + \frac{1}{2}y'Cy$$

are convex functions.

In order to avoid some technical difficulties we assume that for each feasible solution to (2) the gradients of the active constraints are linearly independent.

For fixed y^j, (2) can be partitioned into p subproblems of the form

$$\min\{F_i(x_i) \mid a_\nu'x_i \leq \beta_\nu - b_\nu'y^j, \ \nu = m_{i-1}+1,\dots,m_i\} \tag{16}$$

which can be solved by the algorithm described in Section 3. If the feasible set of (16) has extreme points and if $F_i(x_i)$ is linear we can assume that the optimal solution x_i^j is an extreme point. Then the active constraints can be used to eliminate the x_i variables. If $F_i(x_i)$ is a quadratic function then an optimal solution x_i^j to (16) is in general not an extreme point. In order to eliminate all x_i variables we use appropriate columns of the matrix D_{ij}^{-1} associated with x_i^j.

More generally let x_i^j, $D_{ij}^{-1} = (c_{1i},\dots,c_{n_i i})$, $J_{ij} = \{\alpha_{1i},\dots,\alpha_{n_i i}\}$ be a feasible solution for (16), the associated matrix and index set, respectively, such that

Following 3), it may be necessary to return to 1) and 2). This process may be repeated several times if necessary. It must terminate after a finite number of steps, however. At each application of 1), 2) or 3), the number of α_{ij}'s having value zero is decreased by at least one. Furthermore, application of the algorithm cannot increase the number of these α_{ij}'s. Therefore, after a finite number of steps, a Karush - Kuhn - Tucker point x_j must be obtained with either $\alpha_{ij} \neq 0$ for $i = 1,\ldots,n$, or

$$c_{ij}'Cc_{ij} = 0 \ , \quad \text{for all } i \text{ with } \alpha_{ij} = 0 \ , \tag{14}$$

and

$$c_{kj}'Cc_{\rho j} = 0 \ , \quad \text{for each pair } k,\rho \text{ with } \alpha_{kj} = \alpha_{\rho j} = 0 \ . \tag{15}$$

Assume x_j satisfies the strict complementary slackness condition. We claim that if $\alpha_{ij} \neq 0$ for $i = 1,\ldots,n$, then x_j is a strong local minimizer and if $\alpha_{ij} = 0$ for at least one i then x_j is a weak local minimizer. In the former case, the argument is identical to the proof of Theorem 5. In the latter case, let s be any n - vector with $a_i's = 0$, all $i \in I(x_j)$. From Lemma 2c), there are numbers w_i such that

$$s = \sum_{\alpha_{ij}=-1} w_i c_{ij} + \sum_{\alpha_{ij}=0} w_i c_{ij} \ .$$

From Lemma 2a), 2b), (14) and (15)

$$s'Cs = \sum_{\alpha_{ij}=-1} w_i^2 \geq 0 \ ,$$

from which the assertion follows.

1) If $\alpha_{kj} = 0$ and $c'_{kj}Cc_{kj} > 0$, update using Step 3.1.

2) If $\alpha_{kj} = 0$ and $c'_{kj}Cc_{kj} < 0$, set $s_j = c_{kj}$ and proceed with the algorithm until a new Karush - Kuhn - Tucker point is obtained.

Repeat 1) until $c'_{ij}Cc_{ij} \leq 0$ for all i with $\alpha_{ij} = 0$ then perform 2). Repeat this process until

$$c'_{ij}Cc_{ij} = 0 \quad \text{for all } i \text{ with } \alpha_{ij} = 0 \ .$$

Additional calculations may be required to determine whether or not x_j is a local minimizer.

3) If there are ρ and k with $\alpha_{\rho j} = \alpha_{kj} = 0$, $c'_{\rho j}Cc_{\rho j} = c'_{kj}Cc_{kj} = 0$ and $c'_{\rho j}Cc_{kj} \neq 0$, set

$$s_j = \begin{cases} c_{\rho j} + c_{kj} \ , & \text{if } c'_{\rho j}Cc_{kj} < 0 \\ c_{\rho j} - c_{kj} \ , & \text{otherwise} \ , \end{cases}$$

and proceed with the algorithm using s_j, setting $\alpha_{k,j+1} = \alpha_{\rho,j+1} = 0$ in Step 3.2, until a new Karush - Kuhn - Tucker point is determined.

If s_j is constructed as in 3), then $s'_jCs_j = |2|c'_{\rho j}Cc_{kj} < 0$, and

$$F(x_j - \sigma s_j) = F(x_j) + \sigma^2 s'_jCs_j \ .$$

Thus $F(x_j - \sigma s_j)$ is a strictly decreasing function of σ for all $\sigma \geq 0$ and x_j is not a local minimizer.

$a_i's = 0$, all $i \in I(x_j)$. Then from Lemma 2c), there are numbers w_i such that

$$s = \sum_{\alpha_{ij}=-1} w_i c_{ij} \; .$$

From Lemma 2a),

$$s'Cs = \sum_{\alpha_{ij}=-1} (w_i)^2 \; .$$

At least one of the w_i's must be non-zero. Therefore $s'Cs > 0$ and x_j is indeed a strong local minimizer. We have proved

<u>Theorem 5</u>

Let C be indefinite and let x_o be an extreme point. Then the algorithm terminates in a finite number of steps with either the information that (1) is unbounded from below, or a Karush - Kuhn - Tucker point x_j. In the latter case, if x_j satisfies the strict complementary slackness condition then x_j is also a strong local minimizer for (1).

The assumption that x_o is an extreme point is quite strong. Indeed, R may not possess an extreme point. We now remove the assumption. Let x_o be an arbitrary feasible point and suppose the algorithm has been employed to obtain a Karush - Kuhn - Tucker point x_j. Let D_j^{-1} and J_j be the associated data determined by the algorithm. The property stated in Lemma 4 may not be satisfied. Further calculations must be made in order to either determine that x_j is a local minimizer or find a local minimizer with a better objective function value. Consider the following two steps.

is a quasi - stationary point and the lemma again holds at the next quasi - stationary point. If $\sigma_j = \hat{\sigma}_j$ and the updating proceeds via Step 3.2 then $\alpha_{k,j+1} = 0$ and $g_{j+1}'c_{k,j+1} \neq 0$. Furthermore, $\alpha_{i,j+1} \neq 0$ for all $i = 1,\ldots,n$ with $i \neq k$. At iteration j+1, the algorithm will choose s_{j+1} in Step 1.1, parallel to column k of D_{j+1}^{-1}. Successive iterations will choose the search direction, according to Step 1.1, parallel to the k-th column of the current inverse matrix. Each time the updating uses, Step 3.2 or 3.3, some new constraint becomes active. In at most n steps therefore, either an extreme point is located or the optimal stepsize is used. In the latter case, the k-th column of the new inverse matrix is a conjugate direction. In either case, the next iteration is a quasi - stationary point for which the lemma holds. The assertion of the lemma for all quasi - stationary points now follows by induction.

The finite termination argument of the previous section did not require that C be positive semi - definite. Therefore, with the non - degeneracy assumption of Section 3, the algorithm will terminate in a finite number of steps when C is indefinite. Now consider the case when termination occurs at iteration j. Then x_j satisfies the Karush - Kuhn - Tucker conditions with multipliers

$$u_{\alpha_{ij}} = -g_j'c_{ij} , \quad \text{for all } i \text{ with } 1 \le \alpha_{ij} \le m ,$$

and

$$u_i = 0 , \qquad \text{otherwise} .$$

Assume that x_j satisfies the strict complementary slackness condition; i.e. $u_i > 0$ for all $i \in I(x_j)$. Let s be any non - zero n - vector with

4. The Non - Convex Case

We now consider (1) when C may be indefinite. In this case, (1) may possess several local minimizers and we consider the problem of modifying the algorithm of Section 3 to obtain one. Because C is no longer positive semi - definite, it may occur in Step 2 that $s_j'Cs_j < 0$. But then $F(x_j - \sigma s_j)$ is a strictly decreasing function of σ for all $\sigma \geq 0$ and setting $\tilde{\sigma}_j = +\infty$ is appropriate. Suppose we begin the algorithm with an extreme point x_0 for (1). We will show that with no further modifications, the algorithm will terminate with either a local minimizer or the information that (1) is unbounded from below.

Lemma 4

Let x_0 be an extreme point and let x_j, D_j^{-1} and J_j be determined at the j-th iteration of the algorithm. Then for each quasi - stationary point x_j either $1 \leq \alpha_{ij} \leq m$ or $\alpha_{ij} = -1$ for $i = 1,\ldots,n$.

Proof:

Since x_0 is an extreme point, the lemma is verified for the first quasi - stationary point. We proceed by induction. Assume that x_j is a quasi - stationary point and that the assertion of the lemma holds for it and all previous such points. The algorithm proceeds in Step 1.2 by examining the Lagrange multipliers $u_{\alpha_{ij}} = -g_j'c_{ij}$ for each active constraint α_{ij}. Let k be as in Step 1.2 with $g_j'c_{kj} > 0$. Then $s_j = c_{kj}$. If $\sigma_j = \tilde{\sigma}_j$, then x_{j+1} is a quasi - stationary point, the updating proceeds via Step 3.1, $\alpha_{k,j+1} = -1$ and the lemma holds for the next quasi - stationary point. If $\sigma_j = \hat{\sigma}_j$ and the updating proceeds via Step 3.3, then $\alpha_{k,j+1} \geq 1$, x_{j+1}

for at most n iterations. Suppose then, that s_j is constructed from Step 1.2. From Step 1.1 either there is no i with $\alpha_{ij} = 0$, or,

$$g_j'c_{ij} = 0 \text{ ,} \quad \text{for all i with } \alpha_{ij} = 0 \text{ .} \tag{13}$$

Let $D_j' = [d_{1j},\ldots,d_{nj}]$. By definition of the inverse matrix,

$$g_j = \sum_{i=1}^{n} (g_j'c_{ij})d_{ij} \text{ .}$$

With (12) and (13) this implies

$$g_j = \sum_{1 \le \alpha_{ij} \le m} (g_j'c_{ij})a_{\alpha_{ij}} \text{ ,}$$

which implies that x_j is a quasi-stationary point.

Suppose again that s_j is constructed from Step 1.2. If $\sigma_j > 0$, then $F(x_{j+1}) < F(x_j)$ and the next quasi-stationary point determined will have an objective function value strictly less than $F(x_j)$. Then the associated set of active constraints can never be repeated. Since there are finitely many subsets of the integers 1,2,...,m, termination in a finite number of iterations is assured. If $\sigma_j = 0$, some constraint which is active, but not in the active set, is then added to the active set. This could happen on several consecutive iterations. With the non-degeneracy assumption, however, in no more than n steps all active constraints must be in the active set. Then a strictly positive stepsize must be obtained and the previous argument applies.

$\alpha_{i,j+1} = \alpha_{ij}$,   for all i with $i \neq k$.

Replace j with j+1 and go to Step 1.1.

The critical properties of the matrix D_j^{-1} are summarized in Lemma 2. They are easily proved using Lemma 1 and the Sherman - Morrison formula.

Lemma 2

Let $D_j^{-1} = [c_{1j},\ldots,c_{nj}]$ and $J_j = \{\alpha_{1j},\ldots,\alpha_{nj}\}$ be determined by the algorithm. Then

a) $c_{ij}'Cc_{ij} = 1$, $c_{ij}'Cc_{kj} = 0$ for all i,k with $i \neq k$ and $\alpha_{ij} = \alpha_{kj} = -1$,

b) $c_{ij}'Cc_{kj} = 0$, for all i,k with $0 \leq \alpha_{ij} \leq m$ and $\alpha_{kj} = -1$,

c) $a_{\alpha_{ij}}'c_{kj} = 0$, for all i,k with $1 \leq \alpha_{ij} \leq m$ and $k \neq i$,

d) $a_{\alpha_{ij}}'c_{ij} = 1$, for all i with $1 \leq \alpha_{ij} \leq m$.

Theorem 3

The algorithm terminates in a finite number of steps with either an optimal solution for (1) or the information that (1) is unbounded from below.

Proof:

For each iteration j, it follows from Lemma 2a) and b) and Step 3 that

$$g_j'c_{ij} = 0 \quad \text{for all i with } \alpha_{ij} = -1 . \qquad (12)$$

If s_j is constructed by Step 1.1 then either D_{j+1}^{-1} contains an additional conjugate direction column ($\sigma_j = \tilde{\sigma}_j$) or the number of active constraints increases by one ($\sigma_j = \hat{\sigma}_j$). Therefore, Step 1.1 can be used consecutively

$$(y_j)_i = \begin{cases} a_\ell' c_{ij} , & \text{if } \alpha_{ij} = -1 , \\ 0 , & \text{otherwise} . \end{cases}$$

Set

$$\mu_j = \begin{cases} -1 , & \text{if } (y_j)_\nu > 0 \\ +1 , & \text{otherwise} , \end{cases}$$

$$w_j = \| \mu_j \|y_j\| e_\nu - y_j \|^{-1} (\mu_j \|y_j\| e_\nu - y_j) ,$$

$$p_j = \sum_{\alpha_{ij}=-1} (w_j)_i c_{ij} ,$$

$c_{i,j+1} = c_{ij} - 2(w_j)_i p_j$, for all i with $\alpha_{ij} = -1$ and $i \neq \nu$

$$c_{\nu,j+1} = (a_\ell'(c_{\nu j} - 2(w_j)_\nu p_j))^{-1} (c_{\nu j} - 2(w_j)_\nu p_j) ,$$

$c_{i,j+1} = c_{ij} - (a_\ell' c_{ij}) c_{\nu,j+1}$, for all i with $0 \leq \alpha_{ij} \leq m$,

$\alpha_{i,j+1} = \alpha_{ij}$, for all i with $i \neq \nu$ and $i \neq k$

$\alpha_{\nu,j+1} = \ell$,

$\alpha_{k,j+1} = 0$.

Replace j with j+1 and go to Step 1.1.

<u>Step 3.3:</u> Set

$$c_{k,j+1} = (a_\ell' c_{kj})^{-1} c_{kj} ,$$

$c_{i,j+1} = c_{ij} - \left[\frac{a_\ell' c_{ij}}{a_\ell' c_{kj}}\right] c_{kj}$, for all i with $i \neq k$ and $0 \leq \alpha_{ij} \leq m$,

$c_{i,j+1} = c_{ij}$, for all i with $\alpha_{ij} = -1$,

$\alpha_{k,j+1} = \ell$,

If $\hat{\sigma}_j = \tilde{\sigma}_j = +\infty$, print the message "objective function is unbounded from below" and stop. Otherwise, set $\sigma_j = \min\{\tilde{\sigma}_j, \hat{\sigma}_j\}$ and go to Step 3.

<u>Step 3:</u> <u>Computation of x_{j+1}, D_{j+1}^{-1} and J_{j+1}</u>

Set $x_{j+1} = x_j - \sigma_j s_j$, $g_{i+1} = c + Cx_{j+1}$. Compute $J_{j+1} = \{\alpha_{1,j+1}, \ldots, \alpha_{n,j+1}\}$ and $D_j^{-1} = [c_{1,j+1}, \ldots, c_{n,j+1}]$ as follows. If $\sigma_j = \tilde{\sigma}_j$, then go to Step 3.1 and otherwise go to Step 3.2.

<u>Step 3.1:</u> Set

$$c_{k,j+1} = (c_{kj}'Cc_{kj})^{-1/2} c_{kj} ,$$

$$c_{i,j+1} = c_{ij} - \left[\frac{c_{kj}'Cc_{ij}}{c_{kj}'Cc_{kj}}\right] c_{kj}, \quad \text{for all } i \text{ with } i \neq k, \text{ and } 0 \leq \alpha_{ij} \leq m$$

$$c_{i,j+1} = c_{ij} , \quad \text{for all } i \text{ with } \alpha_{ij} = -1,$$

$$\alpha_{k,j+1} = -1 ,$$

$$\alpha_{i,j+1} = \alpha_{ij} , \quad \text{for all } i \text{ with } i \neq k .$$

Replace j with j+1 and go to Step 1.1.

<u>Step 3.2:</u> If $\alpha_{ij} \geq 0$ for $i = 1, \ldots, n$, then go to Step 3.3. Otherwise, let ν be such that

$$|a_\ell' c_{\nu j}| = \max\{|a_\ell' c_{ij}| \mid \text{all } i \text{ with } \alpha_{ij} = -1\} .$$

If $a_\ell' c_{\nu j} = 0$, go to Step 3.3. Otherwise, for $i = 1, \ldots, n$, set

Step 1: Computation of Search Direction s_j

Let $D_j^{-1} = [c_{1j},\ldots,c_{nj}]$ and $J_j = \{\alpha_{1j},\ldots,\alpha_{nj}\}$

Step 1.1: If there is no i with $\alpha_{ij} = 0$, go to Step 1.2. Otherwise, let k be such that

$$|g_j'c_{kj}| = \max\{|g_j'c_{ij}| \mid \text{all } i \text{ with } \alpha_{ij} = 0\} .$$

If $g_j'c_{kj} = 0$, go to Step 1.2 and otherwise set

$$s_j = \begin{cases} c_{kj}, & \text{if } g_j'c_{kj} > 0, \\ -c_{kj}, & \text{otherwise,} \end{cases}$$

and go to Step 2.

Step 1.2: If there is no i with $1 \le \alpha_{ij} \le m$, then stop with solution x_j. Otherwise, let k be such that

$$g_j'c_{kj} = \max\{g_j'c_{ij} \mid \text{all } i \text{ with } 1 \le \alpha_{ij} \le m\} .$$

If $g_j'c_{kj} \le 0$, then stop with solution x_j. Otherwise set $s_j = c_{kj}$ and go to Step 2.

Step 2: Computation of Stepsize σ_j

Compute $s_j'Cs_j$. If $s_j'Cs_j \le 0$, then set $\tilde{\sigma}_j = +\infty$. Otherwise, set

$$\tilde{\sigma}_j = \frac{g_j's_j}{s_j'Cs_j} .$$

If $a_i's_j \ge 0$ for $i = 1,\ldots,m$, set $\hat{\sigma}_j = +\infty$. Otherwise, compute ℓ and $\hat{\sigma}_j$ such that

$$\hat{\sigma}_j = \frac{a_\ell'x_j - b_\ell}{a_\ell's_j} = \min\left\{\frac{a_i'x_j - b_i}{a_i's_j} \,\middle|\, i \text{ with } a_i's_j < 0\right\} .$$

D_0^{-1} form a set of normalized conjugate directions. We temporarily augment the given problem constraints with the constraints

$$d_i'x = d_i'x_j \ , \quad i = q+1,\ldots,n,$$

and proceed by dropping these before any original problem constraint is dropped.

3. Detailed Formulation of the Algorithm

We now give a detailed statement of the algorithm. The initial data required is a feasible point x_o, an ordered index set $J_o = \{\alpha_{1o},\ldots,\alpha_{no}\}$ and an (n,n) matrix $D_o^{-1} = [c_{1o},\ldots,c_{no}]$. Letting $D_o' = [d_{1o},\ldots,d_{no}]$, the initial data must satisfy $0 \leq \alpha_{io} \leq m$ for $i = 1,\ldots,n$ and for each i with $1 \leq \alpha_{io} \leq m$, we require $a_{\alpha_{io}}' x_o = b_{\alpha_{io}}$ and $d_{io} = a_{\alpha_{io}}$.

At a general iteration j, the algorithm has available x_j, the ordered index set $J_j = \{\alpha_{1j},\ldots,\alpha_{nj}\}$ and the (n,n) matrix $D_j^{-1} = [c_{1j},\ldots,c_{nj}]$. Letting $D_j' = [d_{1j},\ldots,d_{nj}]$, for each i with $1 \leq \alpha_{ij} \leq m$, constraint α_{ij} is active at x_j and its gradient is the i-th column of D_j'. All c_{ij} for which $\alpha_{ij} = -1$ form a set of normalized conjugate directions which are orthogonal to the gradients of the active constraints.

We assume that each $x \in R$ is non-degenerate; i.e., the gradients of those constraints active of x are linearly independent.

With initial data x_o, D_o^{-1} and J_o, we set $j = 0$ and the steps of the algorithm are as follows.

d) The result follows from $\hat{P} = PQ$ and $\hat{P}Q = P(QQ) = P$.

Continuing the previous discussion, we can use

$$P = [c_{q+1},\ldots,c_n] , \qquad d = a_\ell$$

and then apply the lemma to get

$$\hat{P} = [\hat{c}_{q+1},\ldots,\hat{c}_n] .$$

Part a) gives $\hat{c}_i$ in terms of the c_i and a_ℓ, part b) shows that the $\hat{c}_i$ are normalized conjugate directions, and part c) shows that (9) is satisfied. Parts b) and d) show that if D_j' is replaced with

$$D_j' = [a_1,\ldots,a_q,C\hat{c}_{q+1},\ldots,C\hat{c}_n] ,$$

then

$$D_j^{-1} = [c_1,\ldots,c_q,\hat{c}_{q+1},\ldots,\hat{c}_n] .$$

It is possible that $q = n$; i.e., exactly n constraint are active at x_j, or, that a_ℓ is orthogonal to $c_{q+1},\ldots,c_n$. In this case, we obtain D_{j+1}' from D_j' by replacing a_q with a_ℓ. D_{j+1}^{-1} is obtained from D_j^{-1} and a_ℓ using the Sherman - Morrison formula and the conjugate direction columns of D_j^{-1} remain unchanged.

We allow the algorithm to begin with an arbitrary feasible point x_o. Suppose constraints 1,2,...,q are active at x_o. Let

$$D_o' = [a_1,\ldots,a_q,d_{q+1},\ldots,d_n] ,$$

where $d_{q+1},\ldots,d_n$ are any n - vectors such that D_o is non - singular. If $q < n$, we cannot in general assume that the last n - q columns of

Substituting

$$x_i = x_i^j + M_{ij}(y^j - y) \tag{23}$$

into $F_i(x_i)$ and the remaining constraints of (16) we obtain

$$Q_{ij}(y) := F_i(x_i^j + M_{ij}(y^j - y))$$

and

$$a_\nu'(x_i^j + M_{ij}(y^j - y)) \leq \beta_\nu - b_\nu' y$$

or

$$(b_\nu' - a_\nu' M_{ij})y \leq \beta_\nu - a_\nu'(x_i^j + M_{ij}y^j) \; .$$

This leads to the following master problem:

Minimize

$$Q_j(y) := \sum_{i=1}^{p} Q_{ij}(y) + F_0(y)$$

subject to

$$b_\nu' y \leq \beta_\nu \; , \quad \nu = m_p + 1, \ldots, m$$

and

$$(b_\nu' - a_\nu' M_{ij})y \leq \beta_\nu - a_\nu'(x_i^j + M_{ij}y^j)$$

for $i = 1,\ldots,p$ and all $\nu \in \{m_{i-1}+1,\ldots,m_i\}$ such that $\nu \notin J_{ij}$.

Let y^{j+1}, $D_{j+1}^{-1} = (c_{i,j+1},\ldots,c_{n,j+1})$, and $J_{j+1} = \{\alpha_{1,j+1},\ldots,\alpha_{n,j+1}\}$ be an optimal solution to the master problem and the associated matrix and index set, respectively. Define the sets I_j, I_{1j}, ..., I_{pj} such that $r \in I_j$ if and only if $r \in J_{j+1}$ and $r \in \{m_p+1,\ldots,m\}$; $r \in I_{ij}$ if and only if $r \in J_{j+1}$ and $r \in \{m_{i-1}+1,\dots,m_i\}$.

Since gradients of active constraints of (2) are assumed to be linearly independent, it follows that the gradients of the active constraints of the master problem are linearly independent. Thus we may assume that the gradients of all constraints which are active at y^{j+1} are among the columns of D'_{j+1}.

Use (23) to define

$$x_i^{j+1} = x_i^j + M_{ij}(y^j - y^{j+1}) , \quad i = 1,\dots,p.$$

Then it follows from (21) that

$$(x_1^{j+1}, x_2^{j+1}, \dots, x_p^{j+1}, y^{j+1}) \tag{24}$$

is a quasi-stationary point for the problem (2) if there are numbers λ_ν, $\omega_{i\nu}$, and λ_{ir} such that

$$-\nabla F_i(x_i^{j+1}) = \sum_{\nu \in J_{ij}} \omega_{i\nu} a_\nu + \sum_{r \in I_{ij}} \lambda_{ir} a_r, \quad i = 1,\dots,p \tag{25}$$

$$-\nabla F_o(y^{j+1}) = \sum_{\nu \in I_j} \lambda_\nu b_\nu + \sum_{i=1}^{p} \left[\sum_{\nu \in J_{ij}} \omega_{i\nu} b_\nu + \sum_{r \in I_{ij}} \lambda_{ir} b_r \right]. \tag{26}$$

We will first show that (24) is a quasi-stationary point if the following condition is satisfied.

$$a_r \in \operatorname{span}\{a_\nu | \nu \in J_{ij}\} \quad \text{for all } r \in I_{ij},\ i = 1,\dots,p. \tag{27}$$

Indeed, it follows from (22) and (27) that there are numbers $\tau_{i\nu}$ and $\rho_{r\nu}$ such that

$$-\nabla F_i(x_i^{j+1}) = \sum_{\nu \in J_{ij}} \tau_{i\nu} a_\nu \tag{28}$$

$$-a_r = \sum_{\nu \in J_{ij}} \rho_{r\nu} a_\nu \quad \text{for all } r \in I_{ij}, \quad i = 1,\dots,p. \tag{29}$$

Furthermore, because y^{j+1} is an optimal solution to the master problem there are $\lambda_\nu \geq 0$ and $\lambda_{ir} \geq 0$ with

$$-\nabla Q_j(y^{j+1}) = -\sum_{i=1}^{p} \nabla Q_{ij}(y^{j+1}) - \nabla F_o(y^{j+1}) =$$

$$= \sum_{\nu \in I_j} \lambda_\nu b_\nu + \sum_{i=1}^{p} \sum_{r \in I_{ij}} \lambda_{ir}(b_r - M'_{ij} a_r) . \tag{30}$$

Using (29) we have

$$-\sum_{r \in I_{ij}} \lambda_{ir} a_r = -\sum_{r \in I_{ij}} \lambda_{ir} \sum_{\nu \in J_{ij}} \rho_{r\nu} a_\nu =$$

$$= -\sum_{\nu \in J_{ij}} \left(\sum_{r \in I_{ij}} \lambda_{ir} \rho_{r\nu} \right) a_\nu .$$

With

$$\omega_{i\nu} = \tau_{i\nu} + \sum_{r \in I_{ij}} \lambda_{ir} \rho_{r\nu} , \quad \nu \in J_{ij} \tag{31}$$

it follows then from (28) that the equality (25) holds. Observing that

$$-\nabla Q_{ij}(y^{j+1}) = M'_{ij} \nabla F_i(x_i^{j+1})$$

and using (28) and (20) we obtain

$$\nabla Q_{ij}(y^{j+1}) = \sum_{\nu \in J_{ij}} \tau_{i\nu} M'_{ij} a_\nu = \sum_{\nu \in J_{ij}} \tau_{i\nu} b_\nu . \tag{32}$$

By (29) and (20) we have for $i = 1,\dots,p$,

$$-M_{ij}' a_r = \sum_{\nu \in J_{ij}} \rho_{r\nu} b_\nu \qquad \text{for all } \nu \in I_{ij}. \tag{33}$$

Hence, (26) follows from (30) - (33).

In order to compute a quasi - stationary point for problem (2) we, therefore, solve the subproblem (16) and then formulate and solve the master problem. Finally, we use Step 3.2 of the algorithm to update D_{ij}^{-1} by incorporating as many gradients a_r, $r \in I_{ij}$, into D_{ij}' as possible. Denote the new matrix and index set by $D_{i,j+1}^{-1}$ and $J_{i,j+1}$, respectively. If,

$$J_{i,j+1} = J_{ij} \quad \text{for } i = 1,\dots,p \ ,$$

then (27) holds and (24) is a quasi - stationary point. If $J_{i,j+1} \neq J_{ij}$ for at least one i, we use x_i^{j+1}, $D_{i,j+1}^{-1}$, and $J_{i,j+1}$ to formulate and solve a new master problem. Since this iteration strictly increases the total number of positive elements in the index sets, a finite number of iterations suffices to generate a quasi - stationary point.

Now let us assume that (24) is a quasi - stationary point, i.e., equalities (25) and (26) hold. We have seen that $\lambda_\nu \geq 0$ and $\lambda_{ir} \geq 0$. Thus (24) is an optimal solution if $\omega_{i\nu} \geq 0$. Using the properties of D_{ij}^{-1} and D_{j+1}^{-1} we deduce from (28) - (30) that

$$\begin{aligned} \tau_{i\nu} &= -c_{i\nu}' \nabla F_i(x_i^{j+1}) \ , & \nu \in J_{ij}, \quad & i = 1,\dots,p \ , \\ \rho_{r\nu} &= -c_{i\nu}' a_r \ , & \nu \in J_{ij}, \quad & i = 1,\dots,p \ , \\ \lambda_{ir} &= -c_{i,j+1}' \nabla Q_j(y^{j+1}), & r \in I_{ij}, \quad & i = 1,\dots,p \ . \end{aligned} \tag{34}$$

Thus the multipliers $\omega_{i\nu}$ can be computed from (31).

For $i = 1,\dots,p$ determine k_i with

$$\omega_{ik_i} = \min\{\omega_{i\nu} | \nu \in J_{ij}\}$$

and define the set $\hat{I}_j \subset \{1,\dots,p\}$ such that $i \in \hat{I}_j$ if and only if $\omega_{ik_i} < 0$. Then (24) is an optimal solution to (2) if and only if $\hat{I}_j = \emptyset$.

Let $i \in \hat{I}_j$. Then further progress can be made if the constraint with index k_i is dropped from the set of active constraints. Since (21) shows that every constraint with $\nu \in J_{ij}$ is treated like an equality constraint we have to delete the gradient a_{k_i} from the matrix D'_{ij}. If $I_{ij} = \emptyset$, it follows from (31) that $\tau_{ik_i} = \omega_{ik_i} < 0$, which by (34) implies that x_i^{j+1} is not an optimal solution to the subproblem

$$\min\{F_i(x_i) | a'_\nu x_i \le \beta_\nu - b'_\nu y^{j+1},\ \nu = m_{i-1}+1,\dots,m_i\} .$$

Thus we can use the algorithm of Section 3 with initial data x_j^{j+1}, D_{ij}^{-1}, and J_{ij} to compute an optimal solution.

If $I_{ij} \neq \emptyset$, choose any $\ell \in I_{ij}$ and update D_{ij}^{-1} as in Step 3.3 of the algorithm by replacing a_{k_i} in D'_{ij} with a_ℓ.

After modifying the matrix D_{ij}^{-1} in this way for every $i \in \hat{I}_j$ we define and solve a new master problem. Continuing as described above we will obtained a new quasi - stationary point which gives a smaller value of the objective function (2a) than the previous point (24). Thus the method will terminate after a finite number of iterations.

References

[1] O.L. Mangasarian, "Nonlinear Programming", McGraw - Hill, New York, 1969.

[2] B. Noble, "Applied Linear Algebra", Prentice - Hall, Englewood Cliffs, 1969.

[3] J.B. Rosen, "Primal Partition Programming for Block Diagonal Matrices", Numerische Mathematik, 6, 1964, pp. 250 - 260.

[4] J. Sherman and W.J. Morrison, "Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix", The Annals of Mathematical Statistics 20 (1949) p. 621.

Dedicated to George B. Dantzig on the occasion of his seventieth birthday.

URITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 2776	2. GOVT ACCESSION NO. AD-A153 476	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A QUADRATIC PROGRAMMING ALGORITHM		5. TYPE OF REPORT & PERIOD COVERED Summary Report - no specific reporting period
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) M. J. Best and K. Ritter		8. CONTRACT OR GRANT NUMBER(s) DAAG29-80-C-0041 A 81 89 and E 55 82
9. PERFORMING ORGANIZATION NAME AND ADDRESS Mathematics Research Center, University of Wisconsin, 610 Walnut Street, Madison, Wisconsin 53706		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Work Unit Number 5 - Optimization and Large Scale Systems
11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office, P.O. Box 12211, Research Triangle Park, North Carolina 27709		12. REPORT DATE December 1984
		13. NUMBER OF PAGES 29
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Quadratic programming, optimization, conjugate directions, decomposition

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

By using conjugate directions a method for solving convex quadratic programming problems is developed. The algorithm generates a sequence of feasible solutions and terminates after a finite number of iterations. Extensions of the algorithm for nonconvex and large structured quadratic programming problems are discussed.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)